EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES

VIDEO ANALYSIS

Date	16 November 2022
Team ID	PNT2022TMID10687
Project Name	Emerging Methods for Early Detection of Forest Fires

Importing The ImageDataGenerator Library

```
import tensorflow import keras

from keras.preprocessing.image import ImageDataGenerator
```

Define the parameters/arguments for ImageDataGenerator class

```
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,rotati on_range=180,zoom_range=0.2,
horizontal_flip=True) test_datagen=ImageDataGenerator(rescale=1./255)
```

Applying ImageDataGenerator functionality to trainset

```
x_train=train_datagen.flow_from_directory(r'D:/IBM/archive/Dataset/

Dataset/train_set',target_size=(128,128),batch_size=32, class_mode='binary')

Found 436 images belonging to 2 classes.
```

Applying ImageDataGenerator functionality to testset

```
x_test=test_datagen.flow_from_directory(r'D:/IBM/archive/Dataset/
Dataset/test_set',target_size=(128,128),batch_size=32, class_mode='binary')

Found 121 images belonging to 2 classes.
```

Import model building libraries

```
#To define Linear initialisation import Sequential from
keras.models import Sequential #To add layers import Dense from
keras.layers import Dense

#To create Convolution kernel import Convolution2D from
keras.layers import Convolution2D

#import Maxpooling layer

from keras.layers import MaxPooling2D

#import flatten layer from keras.layers import
Flatten import warnings

warnings.filterwarnings('ignore')
```

Initializing the model

model=Sequential() Add CNN Layer

```
model.add(Convolution2D(32,

(3,3),input_shape=(128,128,3),activation='relu'))

#add maxpooling layer

model.add(MaxPooling2D(pool_size=(2,2)))

#add        flatten        layer
model.add(Flatten()) Add Hidden Layer
#add hidden layer

model.add(Dense(150,activation='relu'))

#add output layer

model.add(Dense(1,activation='sigmoid'))
```

Configure the learning process

```
model.compile(loss='binary_crossentropy',optimizer="adam",metrics=["ac curacy"])
```

Train the model

```
model.fit_generator(x_train,steps_per_epoch=14,epochs=10,validation_da ta=x_test,validation_steps=4)

Epoch 1/10

14/14 [==============================] - 41s 3s/step - loss: 1.9923 -

accuracy: 0.6995 - val_loss: 0.2322 - val_accuracy: 0.9256

Epoch 2/10

14/14 [==============================] - 36s 3s/step - loss: 0.7206 -

accuracy: 0.7913 - val_loss: 0.5338 - val_accuracy: 0.8182

Epoch 3/10

14/14 [==============================] - 35s 3s/step - loss: 0.3881 -

accuracy: 0.8647 - val_loss: 0.1472 - val_accuracy: 0.9504

Epoch 4/10

14/14 [==============================] - 36s 3s/step - loss: 0.2811 -

accuracy: 0.8784 - val_loss: 0.0512 - val_accuracy: 0.9835

Epoch 5/10

14/14 [==============================] - 36s 3s/step - loss: 0.2397 -

accuracy: 0.9037 - val_loss: 0.1337 - val_accuracy: 0.9339
```

```
Epoch 6/10

14/14 [==============================] - 36s 3s/step - loss: 0.2102 -

accuracy: 0.9083 - val_loss: 0.0566 - val_accuracy: 0.9917

Epoch 7/10

14/14 [==============================] - 36s 3s/step - loss: 0.1648 -

accuracy: 0.9335 - val_loss: 0.0464 - val_accuracy: 0.9835

Epoch 8/10

14/14 [==============================] - 35s 3s/step - loss: 0.1761 -

accuracy: 0.9220 - val_loss: 0.0440 - val_accuracy: 0.9835

Epoch 9/10

14/14 [==============================] - 36s 3s/step - loss: 0.2224 -

accuracy: 0.9060 - val_loss: 0.0428 - val_accuracy: 0.9917

Epoch 10/10

14/14 [==============================] - 35s 3s/step - loss: 0.1938 - accuracy: 0.9220 - val_loss: 0.0586 -
val_accuracy: 0.9752

<keras.callbacks.History at 0x1ec83f578e0>
```

Save The Model `model.save("forest1.h5")`

Predictions

```
#import load_model from keras.model from
keras.models import load_model #import image class from
keras

from tensorflow.keras.preprocessing import image #import numpy import numpy as np #import
cv2 import cv2

#load the saved model model =
load_model("forest1.h5")

img=image.load_img(r'D:/IBM/archive/Dataset/Dataset/test_set/forest/

0.48007200_1530881924_final_forest.jpg') x=image.img_to_array(img)

res = cv2.resize(x, dsize=(128, 128), interpolation=cv2.INTER_CUBIC)

#expand the image shape

x=np.expand_dims(res,axis=0) pred= model.predict(x)

1/1 [==============================] - 0s 182ms/step pred

array([[0.]], dtype=float32)
```

OpenCV For Video Processing

```
#import opencv library import cv2
#import numpy import numpy as np

#import image function from keras from
keras.preprocessing import image #import load_model
from keras from keras.models import load_model #import
client from twilio API from twilio.rest import Client
#import playsound package

#from playsound import playsound

#load the saved model model=load_model("forest1.h5")
video=cv2.VideoCapture(0) name=['forest','with fire']
```

Creating An Account In Twilio Service

```
account_sid='AC7fbd9e1b65a166f13459d8eca7b664cf'
auth_token='8e7e8e6672a8fb0a908ab3137560022d' client=Client(account_sid,auth_token)
message=client.messages \

.create(

 body='Forest Fire is detected, stay alert', from_='+18434385489', to='+91
95666 05556'

) print(message.sid)

SM60a70f73fc42eacabbbb8f87d34cadbc Sending Alert Message


from tensorflow.keras.utils import load_img,img_to_array while(1):     success, frame=
video.read()     cv2.imwrite("image.jpg",frame)

   img=load_img("image.jpg",target_size=(128,128))     x=img_to_array(img)
x=np.expand_dims(x,axis=0)     predict_x=model.predict(x)

  #classes_x=np.argmax(qqqpredict_x,axis=1)
#pred=model.predict_classes(x)      p=predict_x[0]
print(predict_x)

  #cv2.putText(frame,"predicted class="+str(name[p]),
(100,100),cv2.FONT_HERSHEY_SIMPLEX,1, (0,0,0), 1)
pred=model.predict(x)    if pred[0]==1:
account_sid='AC7fbd9e1b65a166f13459d8eca7b664cf'
auth_token='8e7e8e6672a8fb0a908ab3137560022d'      client=Client(account_sid,auth_token)
message=client.messages \

     .create(

     body='Forest Fire is detected, stay alert', from_='+18434385489',to='+91 95666
05556')       print(message.sid)       print('Fire Detected')       print('SMS sent!')    else:
print('No Danger')       cv2.imshow("image",frame)    if cv2.waitKey(1) & 0xFF ==
ord('q'):       break  video.release() cv2.destroyAllWindows()

1/1 [==============================] - 0s 112ms/step [[0.]]
```

```
1/1 [==============================] - 0s 44ms/step

No Danger

1/1 [==============================] - 0s 30ms/step

[[0.]]

1/1 [==============================] - 0s 26ms/step

No Danger

1/1 [==============================] - 0s 27ms/step

[[1.]]

1/1 [==============================] - 0s 39ms/step

SMd96d46906fc89f8045d0ef2dd63d7c90 Fire Detected

SMS sent!

1/1 [==============================] - 0s 50ms/step

[[1.]]

1/1 [==============================] - 0s 42ms/step

SM3c04ad27dcceb5c97d6338c075ad3639 Fire Detected

SMS sent!

1/1 [==============================] - 0s 32ms/step

[[0.]]

1/1 [==============================] - 0s 31ms/step

No Danger

1/1 [==============================] - 0s 32ms/step

[[1.]]

1/1 [==============================] - 0s 28ms/step

SMa1f8dc905448597cac5146476855ed85 Fire Detected

SMS sent!

1/1 [==============================] - 0s 37ms/step

[[0.01424243]]

1/1 [==============================] - 0s 30ms/step

No Danger

1/1 [==============================] - 0s 36ms/step

[[1.]]

1/1 [==============================] - 0s 29ms/step
```

```
SM3bd2503d7e5e1e2e4021d01f7295fb8b Fire Detected

SMS sent!

1/1 [==============================] - 0s 64ms/step

[[0.]]

1/1 [==============================] - 0s 48ms/step

No Danger

1/1 [==============================] - 0s 43ms/step

[[0.]]

1/1 [==============================] - 0s 49ms/step

No Danger

1/1 [==============================] - 0s 52ms/step

[[1.]]

1/1 [==============================] - 0s 60ms/step

SM3a57c94bccd3c9e0255a263f48f822b0

Fire Detected

SMS sent!
```